media type lists and command event lists, create driver objects, request service objects, and release driver objects.

A1 Contd.

3. The inter-module interface definition of claim 1, wherein the command definition includes at least one of the following service object commands to: release service objects, issue a notice when handling of an event is complete, invoke commands, release work items, suspend work items, resume work items, handle queued events, and cancel queued events.

4. The inter-module interface definition of claim 1, wherein the command definition includes at least one of the following client object commands to: start a work item, release work items, save work item contexts, restore work item contexts, serialize work items, free work item storage, begin batch processing, and end batch processing.

5. A method of inter-module communication comprising:
defining a command definition, wherein
said command definition comprises commands for interfacing with a multi-channel, multi-media, communication queuing system.

6. The method of claim 5 further comprising defining at least one of the following channel driver commands for: requesting media type lists and command event lists, creating driver objects, requesting service objects, and releasing driver objects.

7. The method of claim 5 further comprising defining at least one of the following service object commands for: releasing service objects, issuing a notice when handling of an event is complete, invoking commands, releasing work items, suspending work items, resuming work items, handling queued events, and cancelling queued events.

8. The method of claim 5 further comprising defining at least one of the following client object commands for: starting a work item, releasing work items, saving work item contexts, restoring work item contexts, serializing work items, freeing work item storage, beginning batch processing, and ending batch processing.

A1 cncld.

9. A computer readable storage media comprising:
computer instructions to implement the method of claim 5.

10. A signal in a carrier medium comprising:
computer instructions to implement the method of claim 5.

11. A communication server comprising:
an interface command definition comprising commands for interfacing with one or more communication channel drivers for multiple types of communication media.

12. The communication server of claim 11, wherein the command definition includes a command to start a work item.

A2 cont

13. The communication server of claim 11, wherein the command definition includes a command to release a work item.

14. The communication server of claim 11, wherein the command definition includes a command to save a work item context.

15. The communication server of claim 11, wherein the command definition includes a command to restore a work item context.

16. The communication server of claim 11, wherein the command definition includes a command to serialize a work item.

17. The communication server of claim 11, wherein the command definition includes a command to free work item storage.

18. The communication server of claim 11, wherein the command definition includes a command to begin batch processing.

19. The communication server of claim 11, wherein the command definition includes a command to end batch processing.

20. The communication server of claim 11, further comprising:
a client object operable to interface with the one or more communication channel drivers using at least a portion of the command definition.

21. The communication server of claim 11, further comprising:
a plurality of client objects, wherein each client object interfaces with a service object in one of the communication channel drivers, wherein each service object and each client object correspond to one type of communication media.

A2 cont

22. A channel driver comprising:
an interface command definition comprising commands for interfacing with a multi-channel, multi-media, communication queuing system.

23. The channel driver of claim 22, wherein the command definition includes a command to request a media type list.

24. The channel driver of claim 22, wherein the command definition includes a command to request a command event list.

25. The channel driver of claim 22, wherein the command definition includes a command to create a driver object.

26. The channel driver of claim 22, wherein the command definition includes a command to request a service object.

27. The channel driver object of claim 22, wherein the command definition includes a command to release a driver object.

28. The channel driver of claim 22, wherein the command definition includes a command to issue a notice when handling of an event is complete.

29. The channel driver of claim 22, wherein the command definition includes a command to invoke commands.

30. The channel driver of claim 22, wherein the command definition includes a command to suspend work items.

31. The channel driver of claim 22, wherein the command definition includes a command to resume work items.

A2 cont

32. The channel driver of claim 22, wherein the command definition includes a command to handle queued events.

33. The channel driver of claim 22, wherein the command definition includes a command to cancel queued events.

34. The channel driver of claim 22, wherein the channel driver is operable to interface with a communication server and at least one communication device.

35. The channel driver of claim 34, further wherein the communication server is operable to interface with a queuing system.

36. The channel driver of claim 22, wherein the channel driver is operable to instantiate at least one driver object, wherein the at least one driver object is operable to interface with communication devices for different types of media.

37. The channel driver of claim 36, wherein the at least one driver object is operable to instantiate a service object.

38. The channel driver of claim 37, further wherein each service object includes a task thread to listen for incoming events from a communication device.

39. The channel driver of claim 37, wherein the service object is operable to interface with a communication server, and further wherein the communication server is operable to interface with a queuing system.

40. The channel driver of claim 39, wherein the queuing system is operable to assign work items to agents.

41. The channel driver of claim 22, wherein the commands in the interface command definition are implemented in a data link library.

A2 cont

42. The channel driver of claim 41, wherein commands in the interface command definition are accessed with a function pointer to the data link library.

43. The channel driver of claim 22, further comprising a task thread operable to listen for incoming events.

44. The channel driver of claim 43, wherein the task thread is operable to invoke an event handling function when an event is detected.

45. A method of inter-module communication between at least one channel driver and a communication server, wherein the channel driver is operable to interface with one or more communication devices, and further wherein two or more of the communication devices can use different media types, the method comprising:

defining a command definition, wherein

said command definition comprises commands for interfacing the at least one channel driver with the communication server.

46. The method of claim 45, further comprising:

invoking a command to request a media type list.

47. The method of claim 45, further comprising:
invoking a command to request a command event list.

48. The method of claim 45, further comprising:
invoking a command to create a driver object.

49. The method of claim 45, further comprising:
invoking a command to request a service object.

50. The method of claim 45, further comprising:
invoking a command to release a driver object.

A2 con't

51. The method of claim 45, further comprising:
invoking a command to issue a notice when handling of an event is complete.

52. The method of claim 45, further comprising:
invoking a command to suspend a work item.

53. The method of claim 45, further comprising:
invoking a command to resume a work item.

54. The method of claim 45, further comprising:
invoking a command to handle a queued event.

55. The method of claim 45, further comprising:
invoking a command to cancel a queued event.

56. The method of claim 45, further comprising:
interfacing the communication server with a queuing system.

57. The method of claim 45, further comprising:
detecting incoming events from the communication devices.

58. The method of claim 45, further comprising:
instantiating a task thread to detect incoming events from the communication devices.

59. The method of claim 45, further comprising:
detecting an incoming event from one of the communication devices; and
invoking a function to handle the event.

60. The method of claim 59, further comprising:
queuing the event to a memory cache.

A2 cn't

61. The method of claim 60, further comprising:
indicating the arrival of the event.

62. The method of claim 61, further comprising:
dequeuing the event out of the memory cache and processing the event.

63. A computer readable storage media comprising:
computer instructions to implement the method of claim 45.

64. A signal in a carrier medium comprising:
computer instructions to implement the method of claim 45.

65. An apparatus for inter-module communication comprising:
means for defining a command definition, wherein
said command definition comprises commands for interfacing with a multi-channel, multi-media, communication queuing system.

66. The apparatus of claim 65 further comprising means for defining at least one of the following channel driver commands for: requesting media type lists and

command event lists, creating driver objects, requesting service objects, and releasing driver objects.

67. The apparatus of claim 65 further comprising means for defining at least one of the following service object commands for: releasing service objects, issuing a notice when handling of an event is complete, invoking commands, releasing work items, suspending work items, resuming work items, handling queued events, and canceling queued events.

68. The apparatus of claim 65 further comprising means for defining at least one of the following client object commands for: starting a work item, releasing work items, saving work item contexts, restoring work item contexts, serializing work items, freeing work item storage, beginning batch processing, and ending batch processing.

A2 Con't

Sub B3

69. An apparatus for inter-module communication between at least one channel driver and a communication server, wherein the channel driver is operable to interface with one or more communication devices, and further wherein two or more of the communication devices can use different media types, the method comprising:

means for defining a command definition, wherein

said command definition comprises commands for interfacing the at least one channel driver with the communication server.

70. The apparatus of claim 69, further comprising:

means for invoking a command to request a media type list.

71. The method of claim 69, further comprising:

means for invoking a command to request a command event list.

72. The method of claim 69, further comprising:

means for invoking a command to create a driver object.

73. The method of claim 69, further comprising:

means for invoking a command to request a service object.

74. The method of claim 69, further comprising:
means for invoking a command to release a driver object.

75. The method of claim 69, further comprising:
means for invoking a command to issue a notice when handling of an event is complete.

76. The method of claim 69, further comprising:
means for invoking a command to suspend a work item.

D3
A2
Con'T

77. The method of claim 69, further comprising:
means for invoking a command to resume a work item.

78. The method of claim 69, further comprising:
means for invoking a command to handle a queued event.

79. The method of claim 69, further comprising:
means for invoking a command to cancel a queued event.

80. The method of claim 69, further comprising:
means for interfacing the communication server with a queuing system.

81. The method of claim 69, further comprising:
means for detecting incoming events from the communication devices.

82. The method of claim 69, further comprising:
means for instantiating a task thread to detect incoming events from the communication devices.

83. The method of claim 69, further comprising:

[illegible]
B3
~~A2~~
cncld

means for detecting an incoming event from one of the communication devices;
and
means for invoking a function to handle the event.

84. The method of claim 83, further comprising:
means for queuing the event to a memory cache.

85. The method of claim 84, further comprising:
means for indicating the arrival of the event.

86. The method of claim 85, further comprising:
means for dequeuing the event out of the memory cache and processing the event.